Go With the Reflow
Experiments testing browser rendering

Lindsey Simon
Google UX Developer

June 23, Velocity 2009
4:45 – 5:05pm Regency 2
What is Reflow?

- Everything on the page is really just a box.
What is Reflow?

- “Reflow is the process by which the geometry of the layout engine's formatting objects are computed.”

And that matters because?

- “Reflows are very expensive in terms of performance, and is one of the main causes of slow DOM scripts, especially on devices with low processing power, such as phones. In many cases, they are equivalent to laying out the entire page again.”
What is Reflow?

- CSS rule matching + position recalc + re-paint
- Satoshi Ueyama's tweaked-Firefox3.1 reflow video

http://www.youtube.com/watch?v=AKZZiJo155I
What Can Cause a Reflow?

- Browser resizing
- Setting properties on `element.style` display, padding, margin, font, position, top, left, etc..
- Changes to the live DOM tree appendChild, innerHTML, etc..
- `element.offsetHeight` or `element.offsetWidth`
What Makes Reflows Slow?

- Number of elements
- Depth of the DOM tree
- Number of CSS rules (selector matching)
- CSS rule efficiency (selector matching)
- Type of change
  - display:block vs. visibility:hidden for instance
- Your (user's) browser choice ;)}
- 400 elements, 1000 CSS rules, 2-10 nesting, Different CSS specificities
reflow time by browser

<table>
<thead>
<tr>
<th>DHTML action</th>
<th>Chr1</th>
<th>Chr2</th>
<th>FF2</th>
<th>FF3</th>
<th>IE6,7</th>
<th>IE 8</th>
<th>Op</th>
<th>Saf3</th>
<th>Saf4</th>
</tr>
</thead>
<tbody>
<tr>
<td>className</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
</tr>
<tr>
<td>display none</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1x</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>display default</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>2x</td>
<td>1x</td>
<td>1x</td>
<td>-</td>
<td>1x</td>
<td>1x</td>
</tr>
<tr>
<td>visibility hidden</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>-</td>
<td>1x</td>
<td>1x</td>
</tr>
<tr>
<td>visibility visible</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>-</td>
<td>1x</td>
<td>1x</td>
</tr>
<tr>
<td>padding</td>
<td>-</td>
<td>-</td>
<td>1x</td>
<td>2x</td>
<td>4x</td>
<td>4x</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>width length</td>
<td>-</td>
<td>-</td>
<td>1x</td>
<td>2x</td>
<td>1x</td>
<td>1x</td>
<td>-</td>
<td>1x</td>
<td>-</td>
</tr>
<tr>
<td>width percent</td>
<td>-</td>
<td>-</td>
<td>1x</td>
<td>2x</td>
<td>1x</td>
<td>1x</td>
<td>-</td>
<td>1x</td>
<td>-</td>
</tr>
<tr>
<td>width default</td>
<td>1x</td>
<td>-</td>
<td>1x</td>
<td>2x</td>
<td>1x</td>
<td>1x</td>
<td>-</td>
<td>1x</td>
<td>-</td>
</tr>
<tr>
<td>background</td>
<td>-</td>
<td>-</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>font-size</td>
<td>1x</td>
<td>1x</td>
<td>1x</td>
<td>2x</td>
<td>1x</td>
<td>1x</td>
<td>-</td>
<td>1x</td>
<td>1x</td>
</tr>
</tbody>
</table>

reflow performance varies by browser and action
"1x" is 1-6 seconds depending on browser (1K rules)

http://code.google.com/events/io/sessions/EvenFasterWebsites.html
How Slow?

Reflow Timer: testNonMatchingClass

400 Elements, 1000 CSS Rules (#contents *)
CSS Specificity Matters!
How Slow?

2X Difference in Triton prior to IE8
Webkit is very flat = good
How Slow?

Is the 4X time for the four sides of the box model in IE6,7?
More To Come (soon)!
Speed Up Your Reflows

- Take heavy DOM or Style changes out of the flow
  - Do DOM building/manipulation in document fragments
  - Do DOM building/manipulation with display:none
  - Perform dynamic presentations (animations, dialogs, etc..) in containers that are position:absolute,fixed or otherwise out of the normal document flow
  - Change element.className instead of element.style and Avoid inline styles
    (can prevent multiple procedural reflow if a browser does not do style-change queueing)
  - Trade Smoothness for Speed (the Opera motto)

    e.g. A browser resize listener that resizes at the end of the resize vs. a listener that tries to reflow onresize